Laboratorio di Statistica Aziendale

Introduzione a R

Dott.ssa Michela Pasetto

michela.pasetto2@unibo.it

Installare R

http://www.r-project.org

1. Scegliere il mirror CRAN più vicino
2. Scegliere il proprio sistema operativo
3. Selezionare la versione base
4. Scaricare il programma di installazione

Per installare, eseguire il programma di installazione e seguire le istruzioni.

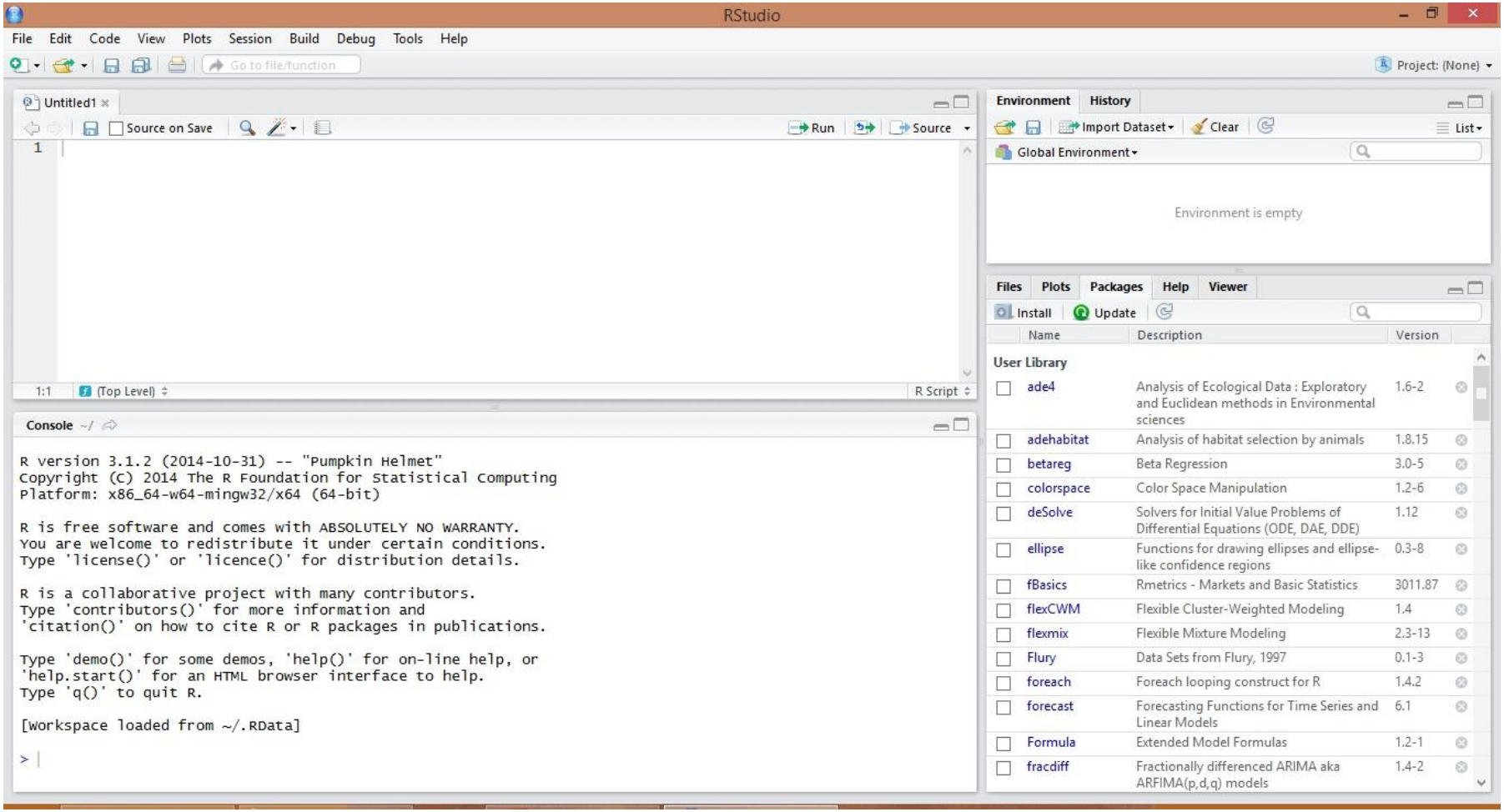
Installare la versione `R Studio`

https://www.rstudio.com/

1. Scaricare l'edizione `R Studio Desktop`

`R Studio` è un'applicazione software che permette di utilizzare `R` in un ambiente più «user-friendly».

Lo Screen di R Studio



La riga di comando nella Console

La riga di comando è identificata dal prompt `>` che indica che l'ambiente è pronto a ricevere istruzioni.

Per eseguire un comando, è sufficiente scriverlo accanto al prompt e premere Invio. Se, quando si preme Invio, il comando non è terminato, risulta un prompt `+`; per correggere, è sufficiente continuare a scrivere sulla riga successiva e poi premere Invio.

Attenzione alla presenza di maiuscole o minuscole, perché `R` è *case sensitive*.

Esempio: utilizzo di R come calcolatrice

Nella riga di comando si possono scrivere delle espressioni, che vengono valutate, e il risultato restituito all'utente:

```
> 5+2
[1] 7
```

Se invece si scrive soltanto:

```
> 2^3 -
```

Sulla Console apparirà il prompt + . Questo indica che il comando digitato non è completo. Il comando può essere completato, per esempio, in questo modo:

```
 > 2^3 -
+ 3
[1] 5
```

Per assegnare un valore ad una variabile si usa <-

```
> Risultato <- 2+3
```

Uscire da R

Dal prompt > si digita il comando `q()` oppure si seleziona con il mouse e si sceglie Exit.

L'uscita dall'ambiente determina la comparsa del messaggio "Salvare area di lavoro?".

L'area di lavoro (o *workspace*) è lo spazio di lavoro che contiene tutti gli oggetti creati con R: i risultati delle elaborazioni, i dati, le variabili e molte altre informazioni. In R Studio il *workspace* è la finestra *environment*, in alto a destra.

All'avvio, R carica un file .RData (il *workspace*) e quando si esce viene data la possibilità di salvare il *workspace* in questo file.

Attenzione alla directory in cui si salva lo spazio di lavoro.

Alcuni comandi per la gestione del *workspace*:

`>source("comandi.R")` legge il file ‘comandi.R’ ed esegue i comandi in esso contenuti;

`>sink("output.txt")` reindirizza i risultati, che di solito appaiono a video, verso il file ‘output.txt’;

`>ls()` elenca gli oggetti contenuti nel workspace corrente;

`>rm()` elimina uno o più oggetti dallo spazio di lavoro;

`>rm(list=ls())` cancella tutti gli oggetti in memoria;

`>save.image()` salva il workspace corrente nel file ".RData";

`>save(x, y, file ="xy.Rdata")` salva solo gli oggetti x e y nel file indicato;

`>load("xy.Rdata")` legge il file indicato e ripristina gli oggetti precedentemente salvati.

Selezionare e modificare la directory

Per cambiare directory, è possibile utilizzare il comando 'Cambia directory...' nel menu File, oppure utilizzare la funzione `setwd()`

```
> setwd("F:/Didattica/Statistica Aziendale")
```

Per conoscere la directory corrente: `getwd()`

```
> getwd()
[1] " F:/Didattica/Statistica Aziendale "
```

Quando si specifica una directory in `R`, si usa lo *slash* (/) invece del *backslash* (quindi è "F:/Didattica/Statistica Aziendale" invece che "F:\Didattica\Statistica Aziendale"). In alternativa, si devono inserire due backslash ("F:\\Didattica\\Statistica Aziendale").

Lo Script

Lo Script è lo spazio nel quale scrivere i codici senza che questi siano eseguiti automaticamente premendo Invio.

Per avere un nuovo Script (in `R Studio`, è la finestra in alto a sinistra), `file > new file > R Script`.

Per eseguire i codici dello Script, posizionarsi con il cursore sulla riga che si desidera eseguire (o selezionare più righe da eseguire) e premere Run (icona nella finestra dello Script in alto a destra) oppure ctrl+R.

Ricordarsi di salvare lo Script, perché è l'unico spazio di `R` in cui si può avere memoria del codice scritto.

Help

Oltre ad alcuni riferimenti bibliografici (es. Iacus, S.M., Masarotto, G. (2006) *Laboratorio di statistica con R*, McGraw-Hill Italia), vi sono manuali disponibili gratuitamente sul sito http://www.r-project.org/ (*Manuals* e *Other – contributed documentation*).

Oppure:

`>?comando`

oppure

`>help(comando)` consente di accedere direttamente all'aiuto in linea di un determinato comando;

`>help.search(argomento)` permette di cercare informazioni su comandi di cui non si conosce il nome o su gruppi di comandi.

I pacchetti

R è una collezione di funzioni che sono aggregate in *packages* (pacchetti) con scopi specifici.

La versione di R scaricata all'installazione è costituta dai *core packages*. Per sapere quali:

```
> getOption("defaultPackages")
```

Installare i pacchetti: i pacchetti non forniti con l'istallazione di R per essere utilizzati devono essere installati via internet (o da file zip locali),

```
> install.packages("spdep")
```

Caricare i pacchetti (attivare i pacchetti installati): quando si carica un pacchetto le sue funzioni diventano utilizzabili ed i suoi dataframe leggibili,

```
> library(spdep)
```

Lettura e scrittura di un file (`read.table` e `write.table`)

`R` può leggere dati registrati in formato testo (ASCII) per mezzo della funzione `read.table()` che ha lo scopo di creare un data frame.

È il modo migliore per leggere i dati in forma di tabella.

```
> salary <- read.table('salary.txt', header=T)
> view(salary)
```

Oppure, se le colonne sono separate da virgole o tabulazioni:

```
> ozone<-read.csv('ozone.csv', header=F, sep=',',
col.names=c('Ozone','Solar.R','Wind','Temp','Month','Day'))
> edit(ozone)
```

Creazione di un oggetto in R

Ciò che è richiesto per salvare un 'qualcosa' che possa essere utilizzato in seguito, in qualsiasi modo, è costruire un **oggetto**. Il termine oggetto, d'ora in avanti, verrà impiegato per indicare qualsiasi cosa in R: espressioni, numeri, formule, insomma TUTTO.

Nello specificare i **nomi degli oggetti** si deve tener conto che ogni nome può essere una sequenza di lettere e numeri purché non inizi con un numero.

Non sono ammessi i seguenti nomi:

`FALSE, TRUE, Inf, NA, NaN, NULL, break, else, for, function, if, in, next, repeat, while.` È logico anche non ridefinire alcune funzioni, come `t, q, c`…

Inoltre, i nomi delle funzioni **non** possono contenere il carattere `_` mentre possono contenere il `.`

L'help di alcuni comandi di R prevede l'uso delle virgolette. Ad esempio, per l'istruzione for:

`>?"for".`

Oggetti e tipologie di dati

In R vi sono diversi tipi di **oggetti**:

- **Vettori**: possono contenere numeri o stringhe;
- **Matrici**: o più in generale array a più dimensioni, sono vettori con due o più indici;
- **Factor**: un tipo di vettore per dati categorici;
- **Liste**: un tipo di vettore generico i cui elementi possono essere di tipo qualunque;
- **Data frames**: strutture tipo matrici in cui le colonne possono essere di tipo diverso;
- **Funzioni**: le funzioni possono essere esse stesse oggetti, essere assegnate ad altre variabili, passate come argomento ad altre funzioni.

Vettori (1)

Un vettore può essere creato tramite la funzione `c()`

```
> y <- c(1,1.2,3.5,4,5)
```

`> 1/y` mostra il reciproco dei numeri contenuti nel vettore y,

```
> y0y <- c(y, 0, y)
> y0y
```

Si può creare un vettore contente valori logici,

```
> nomi <-c('Anna', 'Nunzia', 'Nora', 'Oreste')
```

e concatenare due o più stringhe in una sola

```
> anno <- paste('A', 'N', 'N', 'O', ' 2015',sep='')
```

Vettori (2)

Una sequenza regolare di numeri interi può essere creata nel seguente modo:

```
> X1 <- 1:10
```

La funzione `seq()` crea una sequenza in un range fissato di un passo definito.

```
> X2 <- seq(from=0, to=30, by= 10)
```

L'istruzione `rep()` genera un vettore in cui un oggetto (e, volendo, ogni componente dell'oggetto) viene replicato per un certo numero di volte (rispettivamente, comandi `times` e `each`):

```
> X3 <- rep(1:3, each=3)
> X4 <- rep(x=1, times=5)
```

Nelle espressioni aritmetiche con vettori, le operazioni vengono svolte elemento per elemento,

```
> X5 <- X3-X4
```

Operatori logici

- Maggiore di: `>`
- Minore di: `<`
- Minore o uguale: `<=`
- Maggiore o uguale: `>=`
- Esattamente uguale a: `==`
- Diverso da: `!=`
- x AND y : `x&y`
- x OR y: `x|y`
- NOT x: `!x`
- Test se un vettore x è del tipo desiderato: `is.tipo(x)`

Operazioni con i vettori

```
> x <- c(-1, 2, 3, 4)
> length (x)
[1] 4
> max (x); min(x)
[1] 4
[1] -1
> sum (x);prod(x)
[1] 8
[1] -24
> mean(x)
[1] 2
```

```
> sqrt(x)
[1]   NaN   1.414214   1.732051
2.000000
> x <- c(2,3,4); y <- c(5,6,7)
> x*y
[1] 10 18 28
> x+y
[1] 7 9 11
> x-y
[1] -3 -3 -3
> x+y-3
[1] 4 6 8
> x>y
[1] FALSE FALSE FALSE
```

Vettori - indicizzazione

```
> x <- c(-1, -2, 3.5, 4.5)
> x
> x[3]  # Terzo elemento del vettore
> x[c(1,3)]  # Primo e terzo elemento (indici interi)
> x[c(TRUE, FALSE, TRUE, FALSE)] # Primo e terzo elemento (indici logici)
> x[-2]  # Tutto tranne il secondo elemento
> x[x>0]   # Elementi positivi

> which(x<2)
[1] 1 2
> which ((x>=-1) &(x<5))
[1] 1 3 4
> which ((x <= -2)|(x>1))
[1] 2 3 4
```

I fattori

I fattori possono assumere soltanto valori discreti, definiti come livelli, e possono essere sia qualitativi sia quantitativi.

Esempio:

```
>   vProv   <-   c("BO","RA","FC","RN",   "BO","RA","FC",
"BO","RA","FC", "BO","RA","RA")
> vProv   # è un vettore
 [1] "BO" "RA" "FC" "RN" "BO" "RA" "FC" "BO" "RA" "FC"
"BO" "RA" "RA"
> fProv <- factor(vProv)
> fProv   # è un fattore
 [1] BO RA FC RN BO RA FC BO RA FC BO RA RA
Levels: BO FC RA RN
> summary(fProv)
BO FC RA RN
 4  3  5  1
```

Le matrici (1)

Una matrice è un insieme di elementi dello stesso tipo con due dimensioni. Il comando per generare una matrice è:

```
> A <- matrix(11:16,nrow=3,ncol=2)
> A
     [,1] [,2]
[1,]   11   14
[2,]   12   15
[3,]   13   16
```

Di default, `R` legge le matrici per colonna; il comando `byrow=T` consente di riempire la matrice per riga.

```
> B <- matrix(11:16,nrow=3,ncol=2, byrow=T)
> B
```

Le matrici (2)

```
> t(A)
     [,1] [,2] [,3]
[1,]   11   12   13
[2,]   14   15   16
> sum(A)
[1] 81
> length(A)
[1] 6
> nrow(A)
[1] 3
> ncol(A)
[1] 2
> dim(A)
[1] 3 2
```

Le matrici (3)

Una matrice può anche essere generata unendo più vettori o matrici.

```
> x <- c(11,12,13)
> y <- c(14,15,16)

> cbind(x,y)
      x  y
[1,] 11 14
[2,] 12 15
[3,] 13 16

> rbind(x,y)
  [,1] [,2] [,3]
x   11   12   13
y   14   15   16
```

Le matrici possono essere numeriche, di stringhe di caratteri o logiche.

Matrici - indicizzazione

```
> A[1,2]
[1] 14
> A[1,]  # prima riga
[1] 11 14
> A[,2]  # seconda colonna
[1] 14 15 16

> A==11
      [,1]  [,2]
[1,]  TRUE FALSE
[2,] FALSE FALSE
[3,] FALSE FALSE

> ind <- which((A>11)&(A<=14))
> ind
[1] 2 3 4
> A[ind]
[1] 12 13 14
```

Matrici – operazioni matriciali (1)

Le operazioni matriciali richiedono operatori speciali:

```
> t(A) %*% A  # Prodotto di due matrici
     [,1] [,2]
[1,]  434  542
[2,]  542  677

> A * A       # Prodotto elemento per elemento
     [,1] [,2]
[1,]  121  196
[2,]  144  225
[3,]  169  256

> A + A
     [,1] [,2]
[1,]   22   28
[2,]   24   30
[3,]   26   32
```

Matrici – operazioni matriciali (2)

```
> 2 * A        # Prodotto con uno scalare
     [,1] [,2]
[1,]   22   28
[2,]   24   30
[3,]   26   32

> 2 + A        # Somma con uno scalare
     [,1] [,2]
[1,]   13   16
[2,]   14   17
[3,]   15   18
```

Data frame

Un data frame può essere considerato una matrice le cui colonne contengono dati di tipologia diversa. Le righe del data frame rappresentano le unità statistiche, mentre le colonne rappresentano le variabili rilevate che possono essere sia qualitative (stringhe o valori logici) sia quantitative (numeri): è quella che tipicamente in statistica si definisce **matrice dei dati**.

```
> D <-data.frame(eta=21:24, sesso=c("M","F","F","M"))
> D
   eta sesso
1   21     M
2   22     F
3   23     F
4   24     M
> dim(D)     ## indica la dimensione (numero di casi e variabili)
[1] 4 2
```

Descrizione di un data frame

```
> str(D)                                  # “str” sta per “structure”
'data.frame':   4 obs. of  2 variables:
 $ eta  : int  21 22 23 24
 $ sesso: Factor w/ 2 levels "F","M": 2 1 1 2


> summary(D)
      eta         sesso
 Min.   :21.00   F:2
 1st Qu.:21.75   M:2
 Median :22.50
 Mean   :22.50
 3rd Qu.:23.25
 Max.   :24.00
```

Data frame – indicizzazione (1)

Essendo liste, è possibile accedere alle componenti dei data frame secondo le modalità tipiche delle liste e analogamente a quanto visto per le matrici.

```
> D[[1]]
[1] 21 22 23 24
> D$eta
[1] 21 22 23 24
> D["eta"]
  eta
1  21
2  22
3  23
4  24
```

Data frame – indicizzazione (2)

La funzione `attach()` permette a un database di essere nel *search path* di `R`. Così, si possono digitare direttamente i nomi delle variabili (non si utilizza `$`). Attenzione a non abusare di questo utile comando!

```
> attach(D)
> eta
[1] 21 22 23 24


> D$altezza <- c(175, 164, 170, 182) # Aggiungo la variabile altezza
> D
  eta sesso altezza
1  21     M     175
2  22     F     164
3  23     F     170
4  24     M     182
```

Analisi descrittive univariate: tabelle di frequenza (1)

Esempio di un dataset disponibile in R:

```
> str(warpbreaks)
'data.frame':   54 obs. of  3 variables:
 $ breaks : num  26 30 54 25 70 52 51 26 67 18 ...
 $ wool   : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
 $ tension: Factor w/ 3 levels "L","M","H": 1 1 1 1 1 1 1 1 1 2 ...

> table(warpbreaks$wool)
 A  B
27 27

> table(warpbreaks$wool, warpbreaks$tension)
   L M H
  A 9 9 9
  B 9 9 9
```

Analisi descrittive univariate: tabelle di frequenza (2)

Per suddividere l'insieme delle modalità in classi si può usare la funzione `cut()`:

```
> table(cut(warpbreaks$breaks,
breaks=c(10,20,30,40,50,60,70), include.lowest=T,
right=T))
```

```
[10,20] (20,30] (30,40] (40,50] (50,60] (60,70]
     18      21       6       4       3       2
```

Statistiche descrittive (1)

```
> summary(longley)
```

```
 GNP.deflator        GNP          Unemployed     Armed.Forces
Min.   : 83.00   Min.   :234.3   Min.   :187.0   Min.   :145.6
1st Qu.: 94.53   1st Qu.:317.9   1st Qu.:234.8   1st Qu.:229.8
Median :100.60   Median :381.4   Median :314.4   Median :271.8
Mean   :101.68   Mean   :387.7   Mean   :319.3   Mean   :260.7
3rd Qu.:111.25   3rd Qu.:454.1   3rd Qu.:384.2   3rd Qu.:306.1
Max.   :116.90   Max.   :554.9   Max.   :480.6   Max.   :359.4
   Population         Year         Employed
Min.   :107.6   Min.   :1947   Min.   :60.17
1st Qu.:111.8   1st Qu.:1951   1st Qu.:62.71
Median :116.8   Median :1954   Median :65.50
Mean   :117.4   Mean   :1954   Mean   :65.32
3rd Qu.:122.3   3rd Qu.:1958   3rd Qu.:68.29
Max.   :130.1   Max.   :1962   Max.   :70.55
```

Statistiche descrittive (2)

```
> str(longley)

'data.frame':   16 obs. of  7 variables:
 $ GNP.deflator: num  83 88.5 88.2 89.5 96.2 ...
 $ GNP         : num  234 259 258 285 329 ...
 $ Unemployed  : num  236 232 368 335 210 ...
 $ Armed.Forces: num  159 146 162 165 310 ...
 $ Population  : num  108 109 110 111 112 ...
 $ Year        : int  1947 1948 1949 1950 1951 1952 1953 1954
1955 1956 ...
 $ Employed    : num  60.3 61.1 60.2 61.2 63.2 ...
```

Statistiche descrittive (3)

Le funzioni `cov()` e `cor()`, applicate a più variabili quantitative, calcolano rispettivamente la matrice di varianze e covarianze e quella di correlazione,.

```
> cor(longley)
```

```
             GNP.deflator       GNP Unemployed Armed.Forces Population
GNP.deflator    1.0000000 0.9915892  0.6206334    0.4647442  0.9791634
GNP             0.9915892 1.0000000  0.6042609    0.4464368  0.9910901
Unemployed      0.6206334 0.6042609  1.0000000   -0.1774206  0.6865515
Armed.Forces    0.4647442 0.4464368 -0.1774206    1.0000000  0.3644163
Population      0.9791634 0.9910901  0.6865515    0.3644163  1.0000000
Year            0.9911492 0.9952735  0.6682566    0.4172451  0.9939528
Employed        0.9708985 0.9835516  0.5024981    0.4573074  0.9603906
                     Year  Employed
GNP.deflator 0.9911492 0.9708985
GNP          0.9952735 0.9835516
Unemployed   0.6682566 0.5024981
Armed.Forces 0.4172451 0.4573074
Population   0.9939528 0.9603906
Year         1.0000000 0.9713295
Employed     0.9713295 1.0000000
```

Contiene le stime dei coefficienti di correlazione

Statistiche descrittive (4)

```
> cov(longley)
```

	GNP.deflator	GNP	Unemployed	Armed.Forces	Population
GNP.deflator	116.45763	1063.6041	625.8666	349.0254	73.50300
GNP	1063.60412	9879.3537	5612.4370	3088.0428	685.24094
Unemployed	625.86663	5612.4370	8732.2343	-1153.7876	446.27415
Armed.Forces	349.02537	3088.0428	-1153.7876	4843.0410	176.40981
Population	73.50300	685.2409	446.2742	176.4098	48.38735
Year	50.92333	470.9779	297.3033	138.2433	32.91740
Employed	36.79666	343.3302	164.9103	111.7681	23.46197

	Year	Employed
GNP.deflator	50.92333	36.79666
GNP	470.97790	343.33021
Unemployed	297.30333	164.91027
Armed.Forces	138.24333	111.76811
Population	32.91740	23.46197
Year	22.66667	16.24093
Employed	16.24093	12.33392

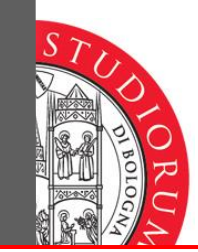
Sulla diagonale principale vi sono le varianze.

Statistiche descrittive (5)

Altra funzione che può essere utilizzata per la costruzione di tabelle è `apply()`. Ad esempio, per calcolare la media delle variabili GNP.deflator, GNP e Unemployed dal data frame "longley" si può usare il seguente comando:

```
> apply(longley [,c("GNP.deflator", "GNP",
"Unemployed")],MARGIN=2,mean)
GNP.deflator          GNP   Unemployed
    101.6813     387.6984      319.3313
```

L'argomento `MARGIN` specifica se applicare la funzione alle colonne (`MARGIN=2`) o alle righe (`MARGIN=1`).



Analisi grafiche

In genere, per creare un grafico, si utilizza `plot()` e quindi `lines()`, `points()`, `legend()`, `text()` e altri comandi per aggiungere annotazioni.

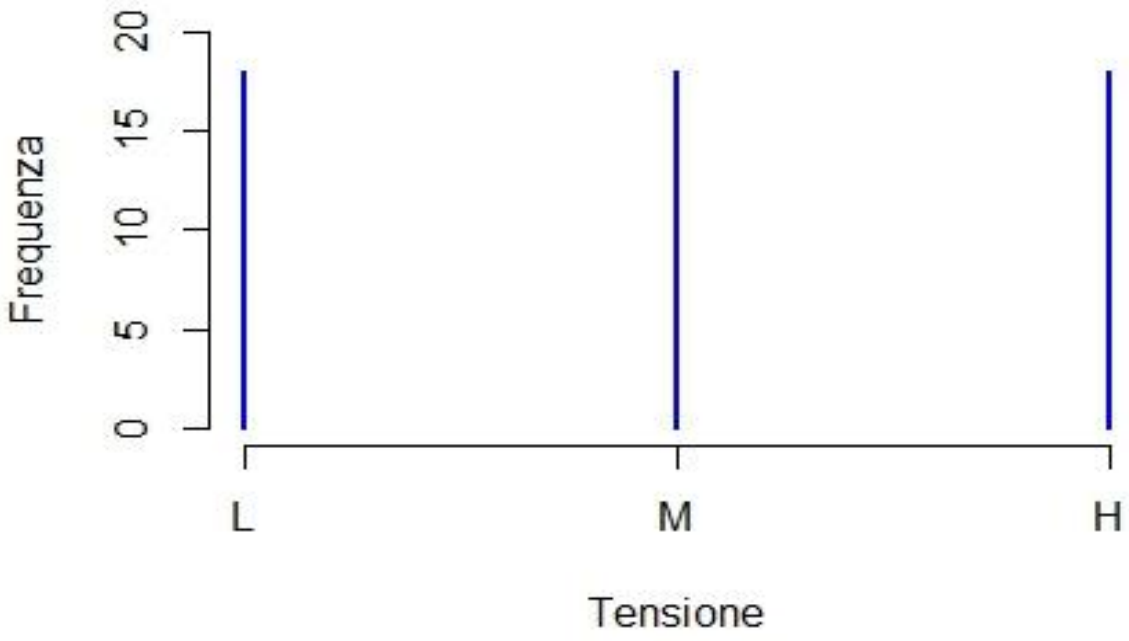
`plot()` è una funzione generica che fa il grafico adatto per tipi di input differenti.

Ci sono due aspetti importanti nella progettazione di un grafico: deve avere qualcosa da dire e deve essere leggibile. Il software può aiutare soltanto nella leggibilità di un grafico.

Rappresentazioni grafiche per variabili qualitative (1)

Diagramma a bastoncini

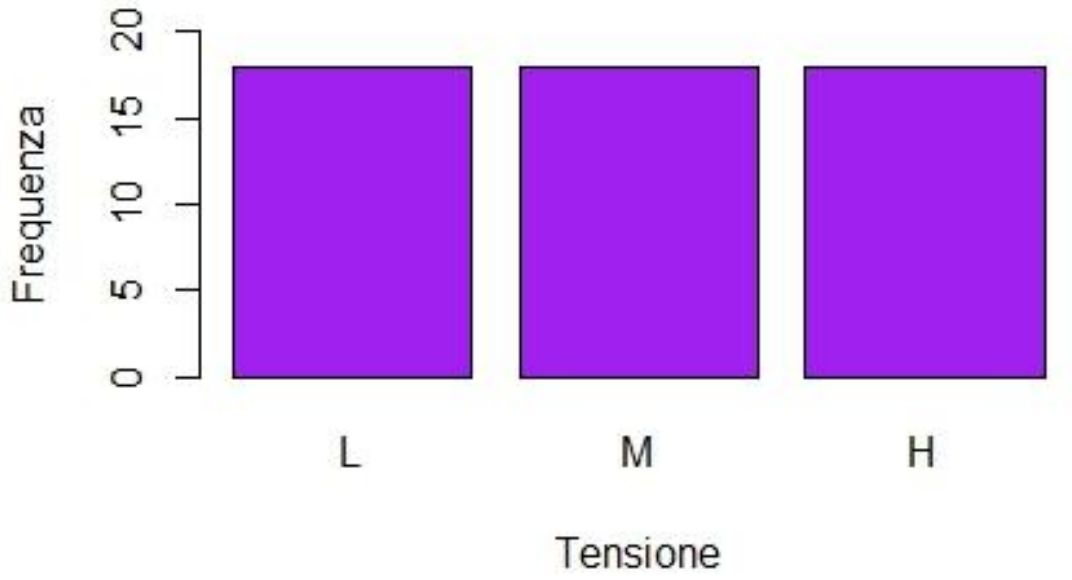
```
> plot(table(warpbreaks$tension), ylim=c(0,20),
ylab="Frequenza", xlab="Tensione", col="blue")
```



Rappresentazioni grafiche per variabili qualitative (2)

Diagramma a barre

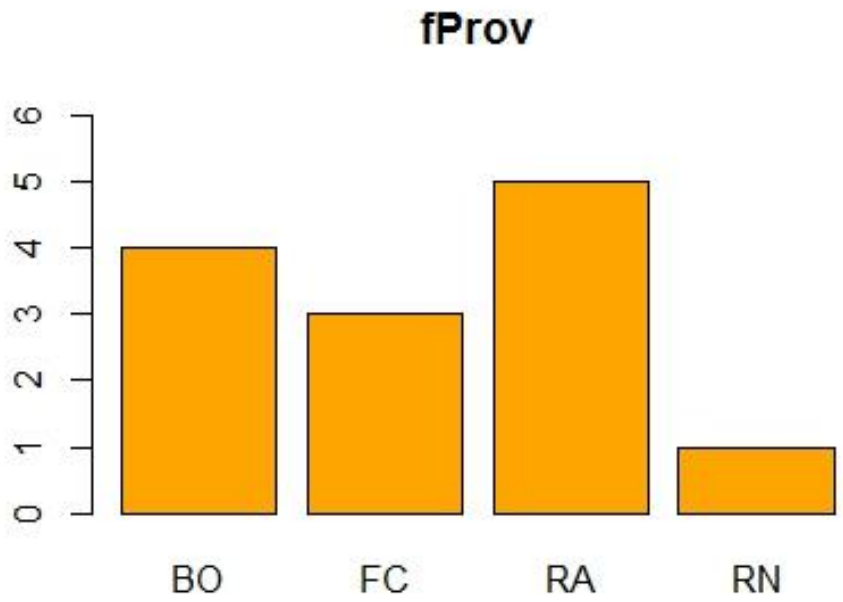
```
> barplot(table(warpbreaks$tension), ylim=c(0,20),
ylab="Frequenza", xlab="Tensione")
```



Rappresentazioni grafiche per variabili qualitative (3)

Se la variabile considerata è un fattore, il comando `plot()` genera automaticamente un grafico a barre:

```
> plot(fProv, col="orange",ylim=c(0,6), main="fProv")
```



Rappresentazioni grafiche per variabili qualitative (4)

Diagramma a torta

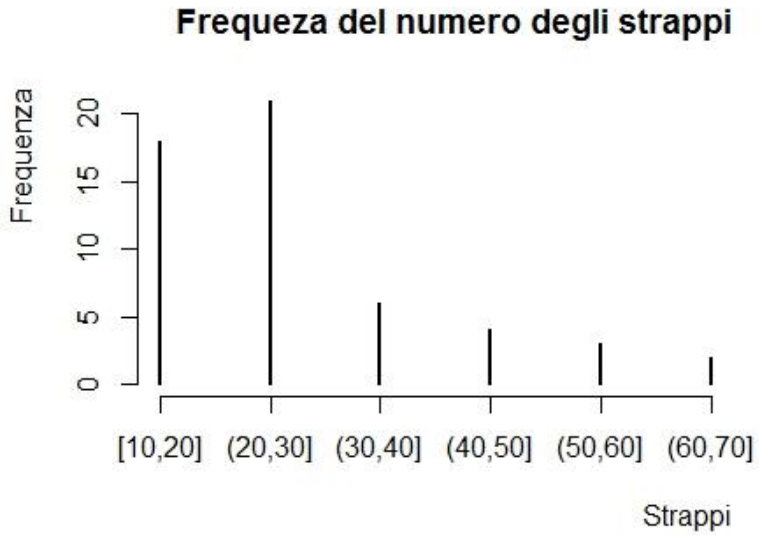
```
> pie(table(warpbreaks$tension), main="Ripartizione della tensione",col=c("green","blue","orange"),radius=1)
```

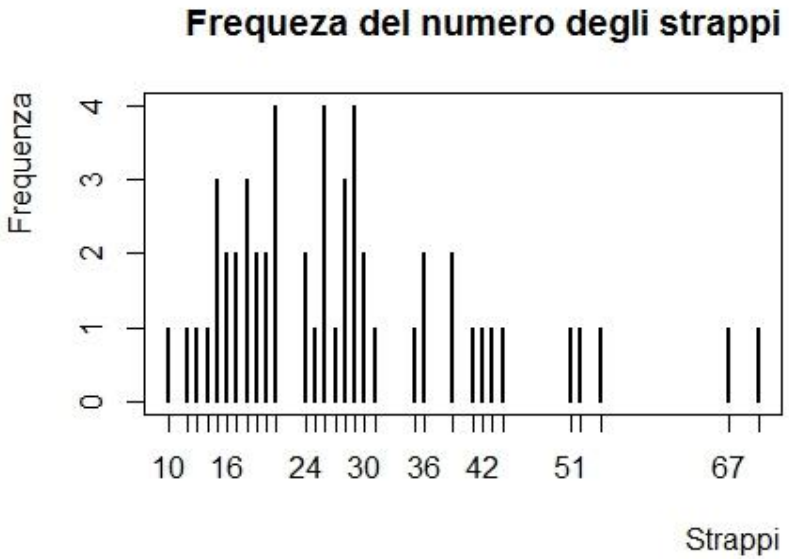
Rappresentazioni grafiche per caratteri quantitativi discreti

Diagramma a bastoncini

```
> plot(table(warpbreaks$breaks), ylab="Frequenza",
xlab="Strappi",main="Numero degli strappi",adj=1)
```

```
> plot(table(cut(warpbreaks$breaks,
breaks=c(10,20,30,40,50,60,70))
```

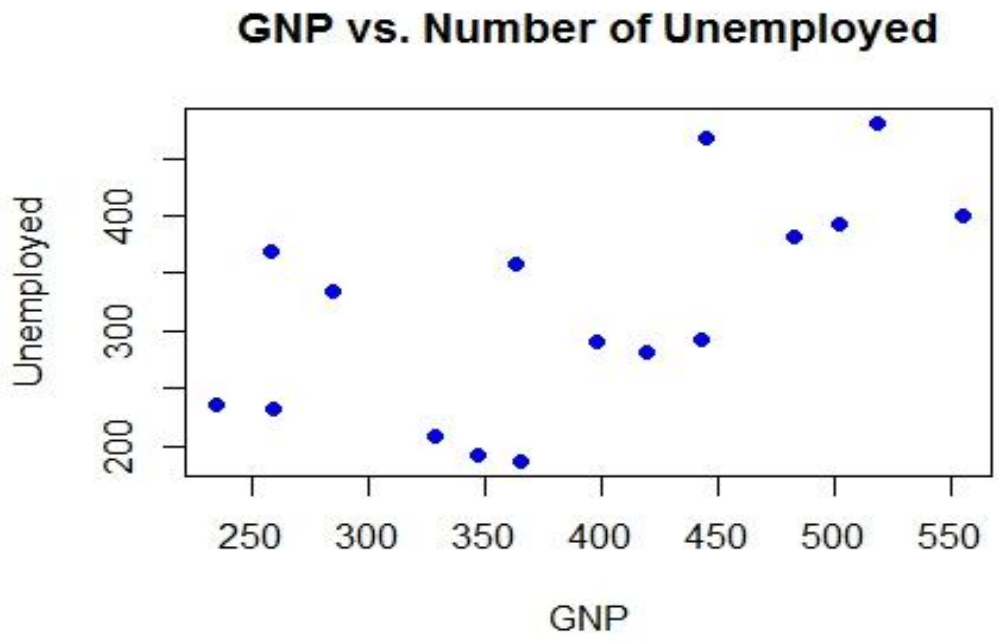




Rappresentazione di due variabili

Scatterplot

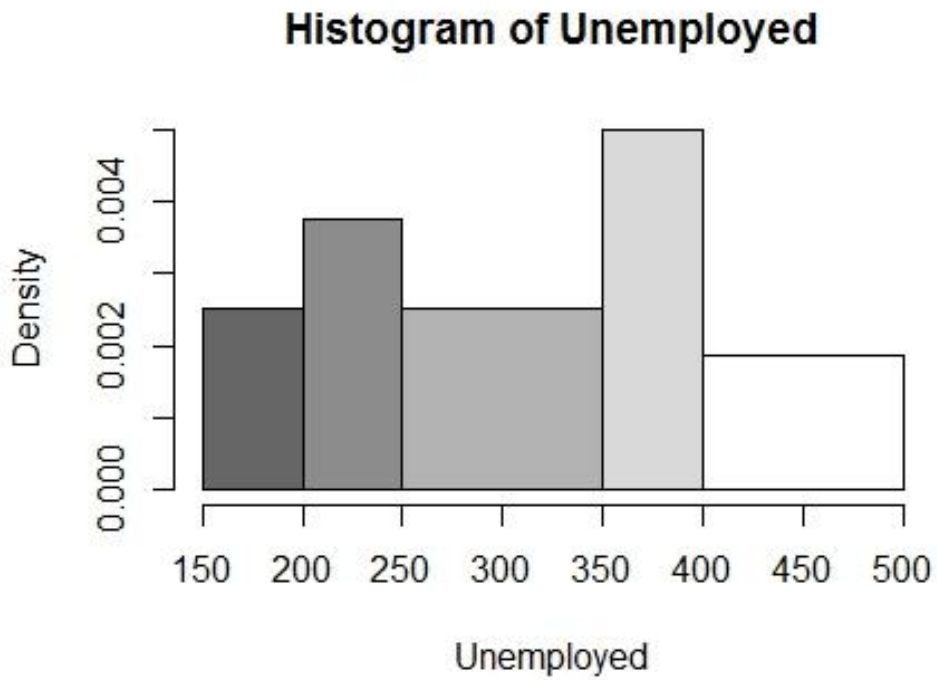
```
> plot(longley$GNP, longley$Unemployed, col="blue",
main="GNP vs. Number of
Unemployed",ylab="Unemployed",xlab="GNP",pch=21,bg="blue")
```



Rappresentazioni grafiche per caratteri quantitativi continui (1)

Istogramma

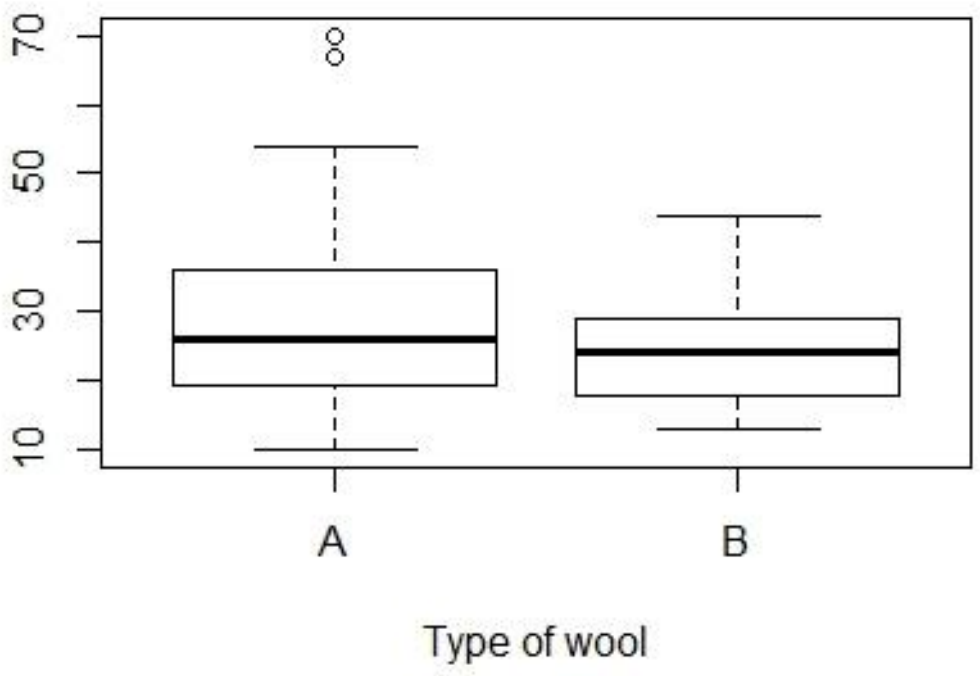
```
> hist(longley$Unemployed,
breaks=c(150,200,250,350,400,500),xlab="Unemployed",col=gr
ay(seq(0.4,1.0,length=5)),main="Histogram of Unemployed")
```



Rappresentazioni grafiche per caratteri quantitativi continui (2)

Boxplot

```
> boxplot(breaks~wool, data=warpbreaks, xlab="Type of wool")
```



Esercizio

Considerare i dataset disponibili in `R chickws` e `ChickWeight`.

Effettuare delle analisi esplorative, tipo medie, varianze, correlazioni, analisi grafiche.

Che cosa si può concludere riguardo le diete alle quali sono sottoposti i polli?